

日 本 国 特 許 庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出 願 年 月 日
Date of Application:

2003年 3月17日

出 願 番 号
Application Number:

特願2003-071867

[ST.10/C]:

[JP 2003-071867]

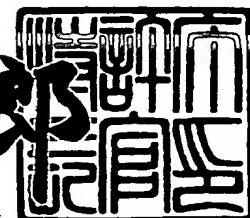
出 願 人
Applicant(s):

ジーイー・メディカル・システムズ・グローバル・テクノロジー・カンパニー・エルエルシー

2003年 4月22日

特 許 庁 長 官
Commissioner,
Japan Patent Office

太田 信一郎



出証番号 出証特2003-3029726

【書類名】 特許願

【整理番号】 16IN-I0224

【提出日】 平成15年 3月17日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 15/00

【発明の名称】 マルチティア・アプリケーション・アーキテクチャ

【請求項の数】 7

【発明者】

 【住所又は居所】 インド国、バンガロール、ホワイトフィールド ロード
 、ジェーエフダブリューティーシー、800 ビジネス
 センター ドライブ

 【氏名】 アービシュカール バーララ

【特許出願人】

 【識別番号】 300019238

 【氏名又は名称】 ジーイー・メディカル・システムズ・グローバル・テク
 ノロジー・カンパニー・エルエルシー

【代理人】

 【識別番号】 100085187

 【弁理士】

 【氏名又は名称】 井島 藤治

【選任した代理人】

 【識別番号】 100090424

 【弁理士】

 【氏名又は名称】 鮫島 信重

【手数料の表示】

 【予納台帳番号】 009542

 【納付金額】 21,000円

【提出物件の目録】

 【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 0005611

【ブルーフの要否】 要

【書類名】 明細書

【発明の名称】 マルチティア・アプリケーション・アーキテクチャ

【特許請求の範囲】

【請求項1】 ミドルティアを有するマルチティア・アプリケーション・アーキテクチャであって、

アプリケーションとミドルティアを仲介するフレームワークを具備し、

前記フレームワークは、

アプリケーションがキャッシュから取り出したオブジェクトをミドルティアに実行させ、

オブジェクトがスタイルしたときそのオブジェクトを予め定められた試行回数の限度内で繰り返しリフレッシュし、

リフレッシュが成功したときはそのオブジェクトをキャッシュに返すとともに再度ミドルティアに実行させ、

オブジェクトのリフレッシュが試行回数の限度内で成功しないときはアプリケーションをフェイルセーフ状態でクイットする、

ことを特徴とするマルチティア・アプリケーション・アーキテクチャ。

【請求項2】 前記試行回数の限度はユーザーにより設定可能である、ことを特徴とする請求項1に記載のマルチティア・アプリケーション・アーキテクチャ。

【請求項3】 前記試行の時間間隔はユーザーにより設定可能である、ことを特徴とする請求項2に記載のマルチティア・アプリケーション・アーキテクチャ。

【請求項4】 前記フレームワークは、その動作がユーザーに対して可視化されている、

ことを特徴とする請求項1ないし請求項3のうちのいずれか1つに記載のマルチティア・アプリケーション・アーキテクチャ。

【請求項5】 前記ミドルティアがクラッシュしたときに正常動作を回復させるウォッチドッグを具備する、

ことを特徴とする請求項1ないし請求項4のうちのいずれか1つに記載のマルチ

ティア・アプリケーション・アーキテクチャ。

【請求項6】 前記ウォッチドッグは定期的なポーリングの結果に基づいてミドルティアを回復させる、
ことを特徴とする請求項5に記載のマルチティア・アプリケーション・アーキテクチャ。

【請求項7】 前記ウォッチドッグは前記フレームワークからの通知に基づいてミドルティアを回復させる、
ことを特徴とする請求項5に記載のマルチティア・アプリケーション・アーキテクチャ。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、マルチティア・アプリケーション・アーキテクチャ (multi-tier application architecture) に関し、とくに、アプリケーション・サーバ (application server) やウェブ・サーバ (Web server) のようなミドルティア (middle tier) を有するマルチティア・アプリケーション・アーキテクチャに関する。

【0002】

【従来の技術】

マルチティア・アプリケーション・アーキテクチャでは、アプリケーション用のビジネスロジック (business logic) は、ミドルティアにおいて実行される。ビジネスロジックはアプリケーションからミドルティアにオブジェクト (object) として指定され、ミドルティアは指定されたオブジェクトを実行する (例えば、非特許文献1, 2 参照)。ミドルティア内のオブジェクトへのアクセス (access) はサービスロケータ (service locator) を通じて行われる (例えば、非特許文献3 参照)。

【0003】

【非特許文献1】

スティーブン グールド (Steven Gould)、" ディベロッ
ップ エヌティア アプリケーションズ ユージング ジェイ2イーイー (De
velop n-tier applications using J2EE
) "、図2、「online」、2002年、ジャバワールド・ドットコム (J
avaWorld.com)、ジャバワールド (Java World)、p.
3/10、「平成15年1月23日検索」、インターネット<URL:http
://www.javaworld.com/javaworld/jw-12
-2000/jw-1201/weblogic_p.html>

【非特許文献2】

リチャード ジー ボールドウィン (Richard G. Bald
win)、" エンタープライズ ジャバビーンズ: ミドルティア サーバース
アンド ジェイ2イーイー (Enterprise JavaBeans:
Middle-Tier Servers and J2EE)"、見出し:ア
ミドルティア サーバ (A Middle-Tier Serer)、「on
line」、2002年、ジュピターメディア コーポレーション (Jupit
ermedia Corporation)、ディベロッパー ドットコム (d
eveloper.com GAMELAN)、p. 4/12-5/12、「平
成14年12月11日検索」、インターネット<URL:http://www
.developer.com/java/other/article.ph
p/641331>

【非特許文献3】

" サン ジャバ センター ジェイ2イーイー パターンズ サービ
ス ロケータ (Sun Java Center J2EE Patterns
Service Locator)、見出し:ソリューション (Soluti
on)、「online」、2002年、サン マイクロシステムズ インコー
ポレーテッド (Sun Microsystems, Inc.)、ジャバテクノ
ロジー ホームページ (Java Techonology Home Pag
e)、p. 3/12-6/12、「平成14年12月24日検索」、インターネ
ット<URL:file:///C:¥WINDOWS¥TEMP¥TD_002

4. DIR¥Sun%20Java%20Center%20-%30Service%20Locator%20J2EE%30Patterns.htm

【0004】

【発明が解決しようとする課題】

上記のようなマルチティア・アプリケーション・アーキテクチャでは、ミドルティア内のオブジェクトへのアクセスがサービスロケータを通じて行われるのでアプリケーションの動作が遅くなる。

【0005】

また、ミドルティアがクラッシュすると、オブジェクトがステイル (stale) することによりアプリケーションがハングアップ (hang-up) するが、そのような場合についてのフェイルセーフ (fail safe) 対策がなされていないので、アプリケーションのハングアップによって、他のアプリケーションへの妨害等、二次的な障害が発生するおそれがある。

【0006】

そこで、本発明の課題は、オブジェクトへのアクセスが高速でかつミドルティアのクラッシュに対するフェイルセーフ機能を備えたマルチティア・アプリケーション・アーキテクチャを実現することである。なお、本書において、アーキテクチャとはコンピュータ・プログラム (computer program) のアーキテクチャを意味する。

【0007】

【課題を解決するための手段】

上記の課題を解決するための本発明は、ミドルティアを有するマルチティア・アプリケーション・アーキテクチャであって、アプリケーションとミドルティアを仲介するフレームワークを具備し、前記フレームワークは、アプリケーションがキャッシュから取り出したオブジェクトをミドルティアに実行させ、オブジェクトがステイルしたときそのオブジェクトを予め定められた試行限度内で繰り返しリフレッシュし、リフレッシュが成功したときはそのオブジェクトをキャッシュに返すとともに再度ミドルティアに実行させ、オブジェクトのリフレッシュが試行限度内で成功しないときはアプリケーションをフェイルセーフ状態でクイッ

トする、ことを特徴とするマルチティア・アプリケーション・アーキテクチャである。

【0008】

本発明では、アプリケーションとミドルティアを仲介するフレームワークにより、アプリケーションがキャッシュから取り出したオブジェクトをミドルティアに実行させ、オブジェクトがスタイルしたときそのオブジェクトを予め定められた試行回数の限度内で繰り返しリフレッシュし、リフレッシュが成功したときはそのオブジェクトをキャッシュに返すとともに再度ミドルティアに実行させ、オブジェクトのリフレッシュが試行回数の限度内で成功しないときはアプリケーションをフェイルセーフ状態でクイットするようにしたので、オブジェクトへのアクセスが高速でかつミドルティアのクラッシュに対するフェイルセーフ機能を備えたマルチティア・アプリケーション・アーキテクチャを実現することができる。

【0009】

前記試行回数の限度はユーザーにより設定可能であることが、ニーズに合致した試行限度とする点で好ましい。前記試行の時間間隔はユーザーにより設定可能であることが、ニーズに合致した試行間隔とする点で好ましい。前記フレームワークは、その動作がユーザーに対して可視化されていることが、ユーザーによる状況把握を容易にする点で好ましい。

【0010】

前記ミドルティアがクラッシュしたときに正常動作を回復させるウォッチドッグを具備することが、回復を自動化する点で好ましい。前記ウォッチドッグは定期的なポーリングの結果に基づいてミドルティアを回復させることが、ウォッチドッグ主導のクラッシュ回復を行う点で好ましい。前記ウォッチドッグは前記フレームワークからの通知に基づいてミドルティアを回復させることが、フレームワーク主導のクラッシュ回復を行う点で好ましい。

【0011】

【発明の実施の形態】

以下、図面を参照して本発明の実施の形態を詳細に説明する。なお、本発明は

実施の形態に限定されるものではない。図1に、マルチティア・アプリケーション・アーキテクチャの構成をブロック (block) 図によって示す。本アーキテクチャは本発明の実施の形態の一例である。本アーキテクチャの構成によって、本発明のマルチティア・アプリケーション・アーキテクチャに関する実施の形態の一例が示される。

【0012】

図1に示すように、本アーキテクチャは、フロントエンドティア (front-end tier) 2、ミドルティア4およびバックエンドティア (back-end tier) 6を有する。

【0013】

フロントエンドティア2はクライアント (client) に相当する。ミドルティア4はアプリケーションサーバに相当する。バックエンドティア6は例えばデータベースサーバ (database server) 等のEIS (enterprise information service) に相当する。なお、フロントエンドティア2とミドルティア4の間に、他のミドルティア例えばウェブサーバ (Web server) 等があってもよい。

【0014】

フロントエンドティア2はバックエンドティア6の資源 (例えば、データベース等) を利用するアプリケーションを有し、そのアプリケーションが必要とするビジネスロジックの実行がミドルティア4によってサービスされる。

【0015】

フロントエンドティア2からミドルティア4へのビジネスロジックの実行要求はオブジェクトとして与えられる。オブジェクトは、フロントエンドティア2内のフレームワーク (framework) 22を通じてミドルティア4に与えられる。

【0016】

図2に、フレームワーク22を含むフロントエンドティア2の主要部の構成をミドルティア4およびバックエンドティア6とともに示す。同図に示すように、フロントエンドティア2は、アプリケーション202およびキャッシュ (cac

h e) 204を有する。

【0017】

キャッシュ204は、アプリケーション202が利用するオブジェクトを保持している。保持するオブジェクトは、アプリケーション202が利用する全てのオブジェクトまたは主要なオブジェクトである。オブジェクトはリモートレファレンス (remote reference) として保持される。

【0018】

アプリケーション202が利用する全てのオブジェクトまたは主要なオブジェクトのリモートレファレンスをキャッシュ204に保持するので、ホームレファレンス (home reference) を用いたリモートレファレンスの (look-up) ルックアップが不要になり、その分アプリケーションの動作は高速になる。

【0019】

フレームワーク22は、ロジックハンドラ (logic handler) 222、ディテクタ (detector) 224、リフレッシャ (refresher) 226およびクイッタ (quitter) 228を有する。

【0020】

アプリケーション202は、キャッシュ204から1つのオブジェクトを取り出してロジックハンドラ222に供給する。ロジックハンドラ222はそのオブジェクトを用いて、ミドルティア4に、対応するビジネスロジックのインボーク (invoke) を行う。

【0021】

ミドルティア4はビジネスロジックを実行する。ビジネスロジックの実効状態および実行結果がディテクタ224によって検出され、ビジネスロジックのサクセス (success) またはフェイル (fail) がロジックハンドラ222に通知される。なお、サクセスはアプリケーション202にも通知される。

【0022】

ビジネスロジックの実行がサクセスした場合は、ロジックハンドラ222は実行済みのオブジェクトをキャッシュに返却し、アプリケーション202は新たな

オブジェクトをキャッシュ204から取り出してロジックハンドラ222に供給する。ビジネスロジックの実行がサクセスする間は、以上のプロセス（process）が繰り返される。

【0023】

ビジネスロジックの実行がフェイルした場合は、オブジェクトがステイルする。このとき、ロジックハンドラ222はリフレッシュャ226にステイルしたオブジェクトのリフレッシュを行わせる。リフレッシュは次のようにして行われる。

【0024】

すなわち、リフレッシュャ226は、オブジェクトのホームレファレンスを用いて、ミドルティア4に対してリモートレファレンスのルックアップを行う。

【0025】

ルックアップが成功したときは、ミドルティア4からリフレッシュャ226にリモートレファレンスが返される。これによって、ステイルしたオブジェクトのリフレッシュが行われる。そこで、ロジックハンドラ222はリフレッシュされたオブジェクトをキャッシュ204に返すとともに、そのオブジェクトについてあらためてインボークを行う。これによって、ステイルしたオブジェクトを復活させることが可能になる。

【0026】

ルックアップが不成功のときはリモートレファレンスが帰ってこないので、ルックアップのやり直しが行われる。ルックアップの不成功が続く間、ルックアップのやり直しが行われる。すなわち、リフレッシュの試行が複数回繰り返される。これによって、オブジェクトのリフレッシュの成功率が向上する。

【0027】

繰り返しの回数の上限および繰り返しの時間間隔は予め定められている。回数および間隔はユーザー（user）が設定可能にすることが、ニーズに合致した回数および間隔とする点で好ましい。

【0028】

繰り返し回数が上限に達してもリフレッシュが成功しないときは、ロジックハンドラ222はクイッタ226にフェイルセーフを行わせる。クイッタ226は

、アプリケーション202が専有していた各種の資源を解放すること等により、アプリケーション202についてフェイルセーフクイット (fail safe quit) を行う。このようなフェイルセーフ処理により、その後のアプリケーション202の再起動を円滑にすることができ、また、資源を共用する他のアプリケーションへの妨害をなくすることができる。

【0029】

このようなフレームワーク22は、エンティティビーンズ (entity beans) やステイトレス (stateless) のセッションビーンズ (session beans) のハンドリングにとくに好適である。

【0030】

以上のようなフレームワークの動作状況は、ユーザーが認識可能なものとされる。これは、適宜のGUI (graphical user interface) 等を利用して行われる。これによって、ユーザーによる状況把握を容易にすることができる。なお、必要がなければこれは省略してもよい。

【0031】

ミドルティア4におけるビジネスロジックのフェイルはミドルティア4のクラッシュ (crash) 等によって発生する。クラッシュが発生した場合でもそれを自動的に回復させることができれば、ミドルティア4の稼働率を良くすることができ、アプリケーションを効果的に実行することが可能になる。

【0032】

そこで、図3に示すように、ウォッチドッグ (watchdog) 402を用いてミドルティア4のクラッシュの自動回復を行う。ウォッチドッグ402はポーリング (polling) を定期的に行ってミドルティア4のクラッシュの有無を監視し、クラッシュが発生したときはその回復作業を行う。このようなウォッチドッグ402は、例えばシェルスクリプト (shell script) 等を用いて実現することができる。

【0033】

ウォッチドッグ402によるミドルティア4の回復は、図4に示すように、ロジックハンドラ222からのクラッシュ通知に基づいて行うようにしてもよい。

このようなウォッチドッグ 4 0 2 は、専用のアプリケーション等によって実現することができる。なお、ミドルティア 4 のクラッシュは、ロジックハンドラ 2 2 2 がリフレッシュによるオブジェクトの復活ができなかったときに通知される。

【 0 0 3 4 】

【発明の効果】

以上詳細に説明したように、本発明によれば、オブジェクトへのアクセスが高速でかつミドルティアのクラッシュに対するフェイルセーフ機能を備えたマルチティア・アプリケーション・アーキテクチャを実現することができる。

【図面の簡単な説明】

【図 1】

本発明の実施の形態の一例の構成を示すブロック図である。

【図 2】

本発明の実施の形態の一例についてフロントエンドティアの構成をより詳細に示すブロック図である。

【図 3】

本発明の実施の形態の一例についてフロントエンドティアの構成をより詳細に示すブロック図である。

【図 4】

本発明の実施の形態の一例についてフロントエンドティアの構成をより詳細に示すブロック図である。

【符号の説明】

- 2 フロントエンドティア
- 2 2 フレームワーク
- 4 ミドルティア
- 6 バックエンドティア
- 2 0 2 アプリケーション
- 2 0 4 キャッシュ
- 2 2 2 ロジックハンドラ
- 2 2 4 ディテクタ

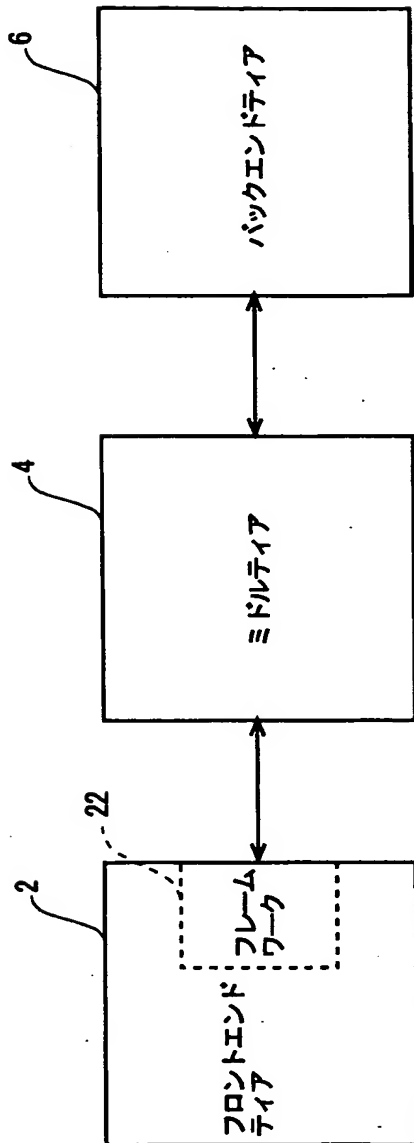
226 リフレッシュ

228 クイッタ

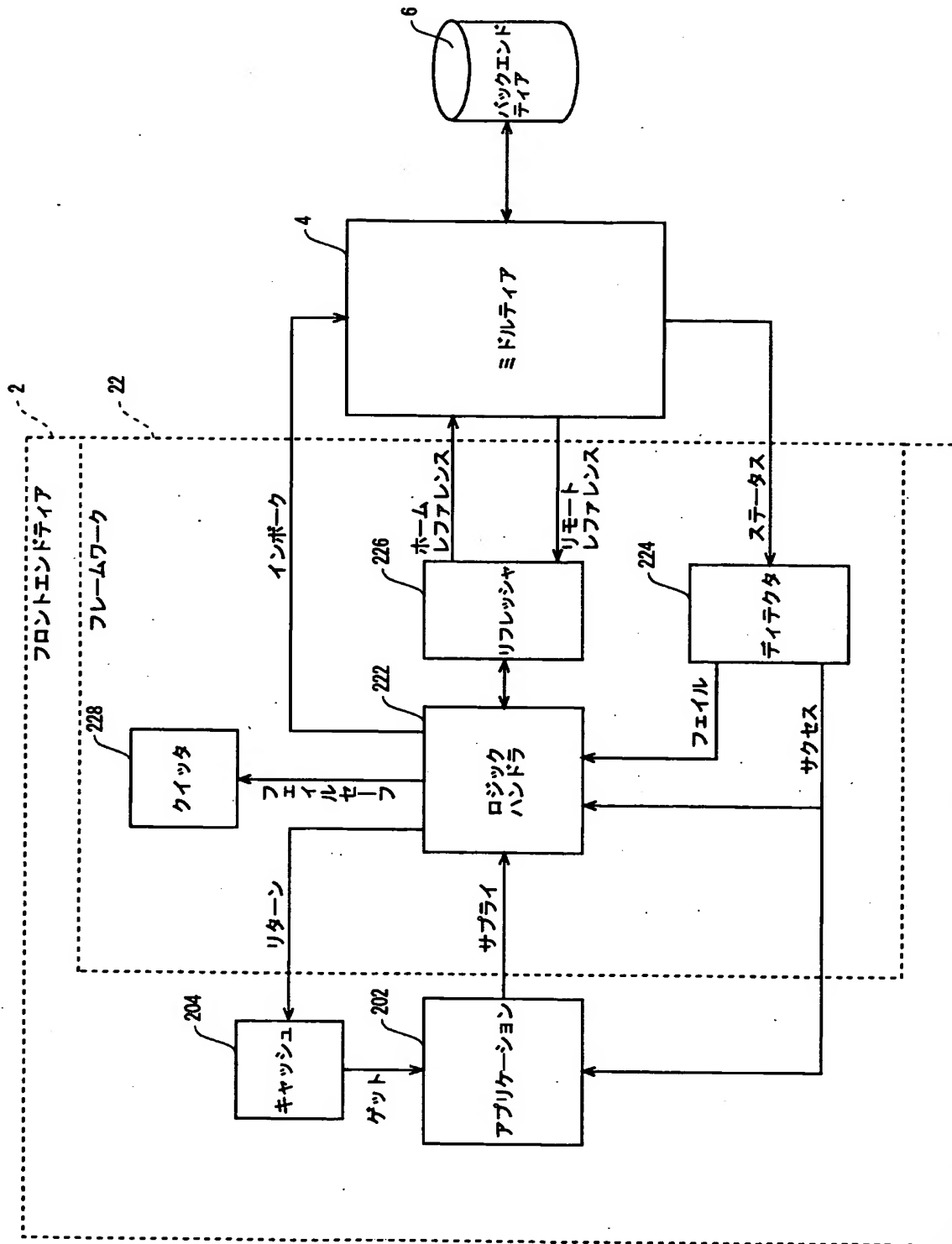
402 ウォッチドッグ

【書類名】 図面

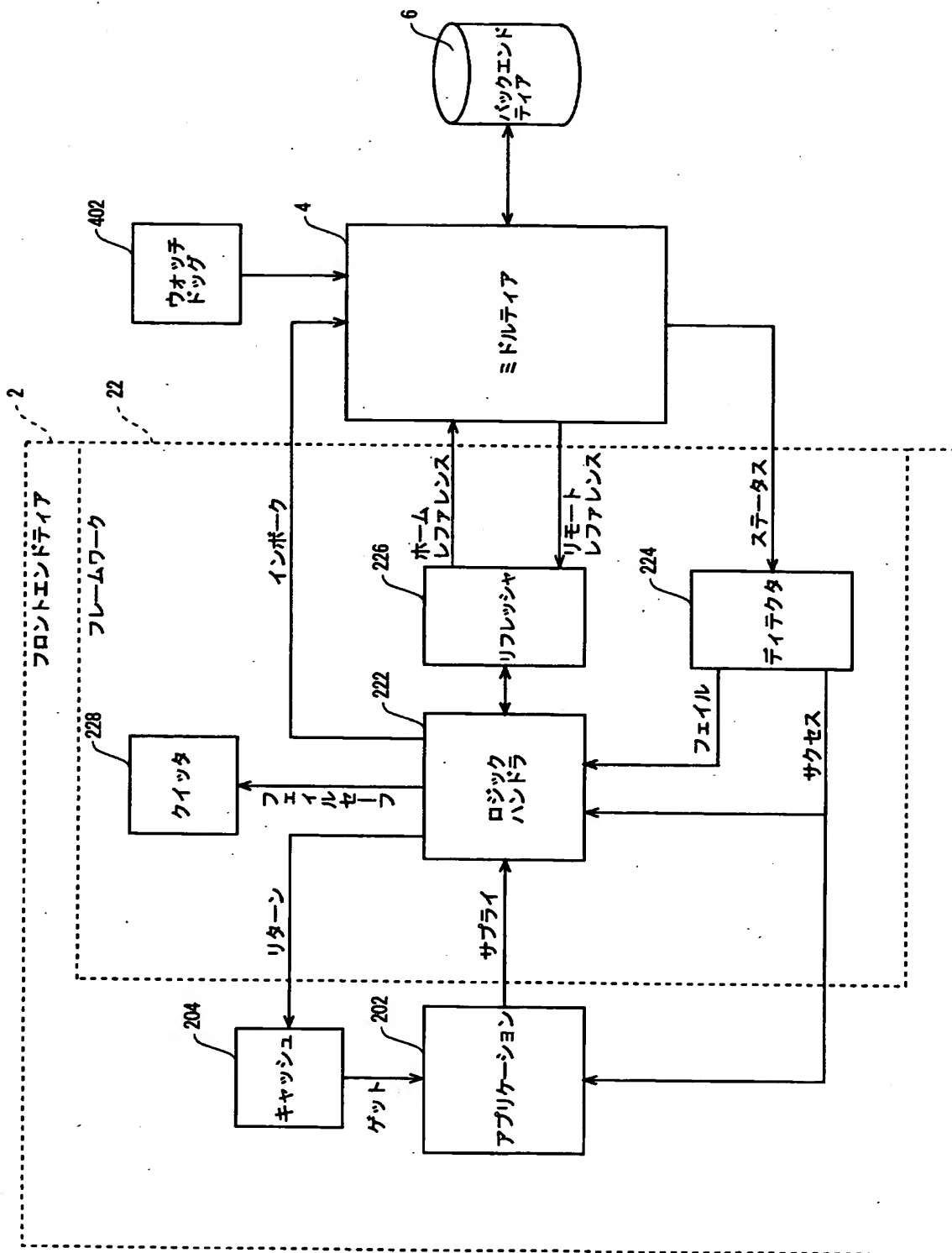
【図 1】



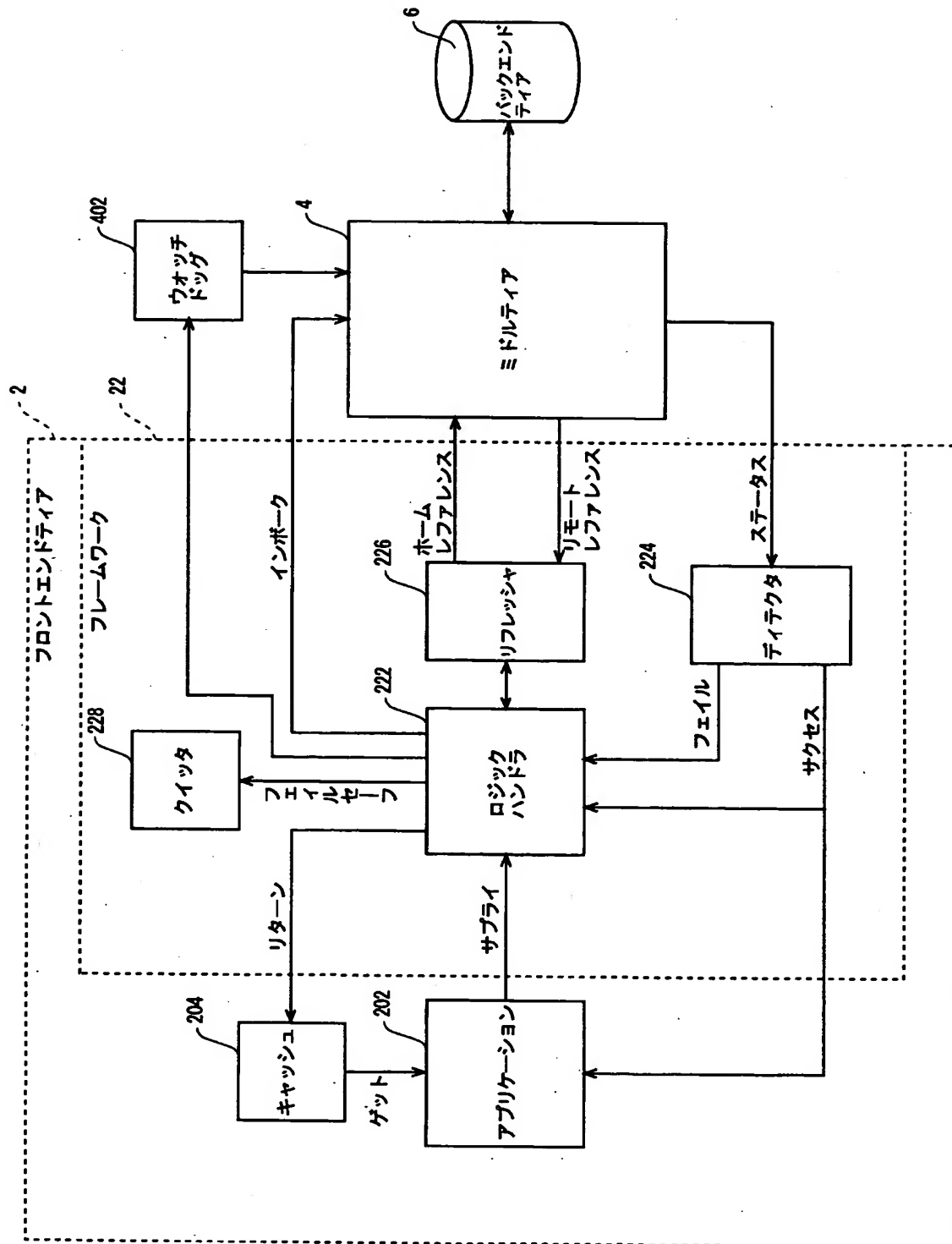
【図 2】



【図 3】



【図4】



【書類名】 要約書

【要約】

【課題】 オブジェクトへのアクセスが高速でかつミドルティアのクラッシュに対するフェイルセーフ機能を備えたマルチティア・アプリケーション・アーキテクチャを実現する。

【解決手段】 アプリケーション（202）とミドルティア（4）を仲介するフレームワーク（22）により、アプリケーションがキャッシュ（204）から取り出したオブジェクトをミドルティアに実行させ（222）、オブジェクトがステイルしたときそのオブジェクトを予め定められた試行回数の限度内で繰り返しリフレッシュを試行し（226）、リフレッシュが成功したときはそのオブジェクトをキャッシュに戻すとともに再度ミドルティアに実行させ、オブジェクトのリフレッシュが試行回数の限度内で成功しないときはアプリケーションをフェイルセーフ状態でクイットする（228）。

【選択図】 図2

出 願 人 履 歴 情 報

識別番号 [300019238]

1. 変更年月日 2000年 3月15日

[変更理由] 名称変更

住 所 アメリカ合衆国・ウィスコンシン州・53188・ワウケシャ
・ノース・グランドビュー・ブルバード・ダブリュー・71
0・3000

氏 名 ジーイー・メディカル・システムズ・グローバル・テクノロジー
・カンパニー・エルエルシー